What You Should Get with GIT

James Conroy

St Louis Unix Users Group

July 13, 2022

What is it

A version control system

what is version control

A way to track changes to files over time.

Advantages of version control

- you know who made the change
- you know why the change was made
- you can go back to before a change was made
- you can inspect changes in isolation
- No more “file.name” “file.name.bak” “file.name.bak-good” “file.name-good-version”

Why use git instead of another solution

The branching model.

Git allows you to have multiple independent versions of your repo that can switched between easily.

Frictionless Context Switching

Create a branch to test an idea, switch to another branch to address a bug, return to your test branch.

Role-Based Branches

You can have a branch as a single source of truth for what is currently in production, another for code being tested and any number of branches for day to day work.

Feature Based Workflow

> Create new branches for each new feature you're working on so you can seamlessly switch back and forth between them, then delete each branch when that feature gets merged into your main line.
>
> – *www.git-scm.com/about*

Getting GIT

- git bash on windows
- it's in your package manager
- failing that you can get it from it's website

How do you use git

Couldn't be simpler

```
$ mkdir project
$ git init

$ cat > foobar.md << EOF
# this is a file

this is a paragraph
EOF

$ git add foobar
$ git commit
```

You have now saved a snap shot of your code.

branches

How do you use them

```
# to list branches
git branch

# to switch to a branch
git switch <branch name>

# to create a branch and then switch to it
git switch -c <branch name>
```

Merging

When you're done with your changes you can `merge` them into another branch

```
git switch master
```

```
git merge <branch name>
```

Collaboration

There are a lot of ways to use git to manage source code and collaboration, from GitHub's model of forks and pull requests to sending patches over email to mailing lists.

Code forges

- BitBucket
- CodeBurg
- GitHub
- GitTea
- GitLab
- SourceHut
- Other self hosted Solutions
- Mailing Lists (etc Linux Kernel, Git)

Each has a different workflow, with GitHub being the most popular. Generally you make a repo on the website and then push your repo up to it.

Code forges

All the solutions have instructions on how to set up repos with them, so we're going to assume you have a remote repo to work with.

Working with Remotes

```
# display your repos remote names and urls
git remote -v
# add a new remote
git remote add origin <url>

# get changes from the remote repository
git pull

# push your changes to the remote repository
git push
```

When working with others it is a good idea to perform a pull first before pushing

Working with Remotes

Git pull and push will operate on the current branch unless told otherwise

```
git push <remote name> <branch name>
```

You can push local branches to the remote by

```
git push --set-upstream <remote name> <branch-name>
```

More Resources

- git-scm.com/book/en/v2
- git-send-email.io
- jwiegley.github.io/git-from-the-bottom-up/